

PeVEL

A Promise-Based Civilizational Engine

Book B

A Technical Specification for Recording Commitments Without Reduction,

Exchange Without Coercion, and Trust Without Scoring

By Young Keun Choi

With ChatGPT • Claude • Gemini

This book is a specification.

It describes a system boundary, not a worldview.

It proposes no reform, predicts no future, and claims no authority.

Book B of the *One Person State* series

This document defines what a system can do under constraints it does not choose.

Table of Contents

Technical Preface

Scope, Assumptions, and Non-Goals

Book A Linkage: Entire Book A as inherited constraint

Part I — From Recognition to Formalization

Why Human Promises Break at Scale

Chapter 1. The Limits of Human Memory and Attention — *Why recognition alone fails beyond small groups*

Book A §1: The Contract You Never Signed Book A §2: Why We Built the Cage

Chapter 2. Time, Drift, and the Loss of Shared Context — *Why promises decay without formal trace*

Book A §8: Society as a Network of Promises

Chapter 3. When a Promise Becomes Unjudgeable — *The minimum conditions required for outcome determination*

Book A §6: The Three Axioms Book A §7: Why Contract Is Not Promise

Part II — Promise as an Atomic Unit

Defining the Smallest Formal Commitment

Chapter 4. Why Transactions Are the Wrong Primitive — *Structural limits of exchange-based models*

Book A §7: Why Contract Is Not Promise Book A §4: Money as a Crude Compression of Human Value

Chapter 5. The Promise Atom — *Scope, Counterparty, Time Window, Completion Condition*

Book A §6: The Three Axioms

Chapter 6. States of a Promise — *Declared / Fulfilled / Unfulfilled / Withdrawn*

Book A §8: Society as a Network of Promises

Part III — Valuation Without Reduction

Recording Change Without Compressing Value

Chapter 7. Why Scalar Value Collapses Meaning — *Technical limits of single-metric systems*

Book A §4: Money as a Crude Compression of Human Value

Chapter 8. Δ State: Recording Change, Not Worth — *State transition as the only calculable object*

Book A §6: The Three Axioms (Non-violation)

Chapter 9. Multi-Dimensional Accumulation — *Partial order, local context, non-fungibility*

Book A §11: The Hidden Price of Mistrust

Part IV — Exchange Without Coercion

How Interaction Occurs Without Control

Chapter 10. Why Enforcement Is a Systemic Shortcut — *What coercion solves, and what it destroys*

Book A §10: When Trust Is Replaced by Control

Chapter 11. Voluntary Matching — *Mutual opt-in as a system primitive*

Book A §9: Why Trust Cannot Be Forced

Chapter 12. Refusal, Exit, and Non-Participation — *Refusal as first-class state, exit without penalty*

Book A §12: Why Ethics Cannot Be Engineered

Part V — Trust as an Emergent Property

How Trust Affects the System Without Becoming a Score

Chapter 13. Why Trust Cannot Be Recorded Directly — *Distinction between trust and reputation*

Book A §9: Why Trust Cannot Be Forced

Chapter 14. τ as Potential, Not Measure — *Contextual visibility, decay, and recovery*

Book A §11: The Hidden Price of Mistrust

Chapter 15. System Effects of Accumulated Trust — *Constraint relaxation without ranking or reward*

Book A §15: From Individual Choice to Collective Shift

Part VI — System Boundaries & Parallel Coexistence

Where the Engine Stops

Chapter 16. Opt-In by Design — *Participation without obligation*

Book A §13: Why This Is Not a Revolution

Chapter 17. No Override, No Replacement — *Legal, monetary, and political non-interference*

Book A §14: Parallel Coexistence with Existing Systems

Chapter 18. Graceful Failure — *Irrelevance over coercion*

Book A §16: OPS as a Civilizational Protocol

Appendices (*Non-Canonical*)

Appendix A: Formal Definitions Appendix B: Failure Modes

Technical Preface

Scope, Assumptions, and Non-Goals

This is a technical document.

It does not persuade. It does not introduce philosophy. It does not argue for a future.

Those tasks belong elsewhere, and have been completed there.

What follows is a specification: the scope of a system, the assumptions it inherits, the constraints it obeys, and the boundaries it does not cross.

Readers seeking meaning, justification, or moral reasoning will not find them here. Readers seeking a mechanism that functions under strict constraints may proceed.

1. Scope

This book describes a promise-based engine for recording and coordinating human commitments.

It specifies:

- how promises are represented as formal objects,
- how their outcomes are recorded without collapsing into a single measure,
- how exchange occurs without coercion,
- how trust affects future interactions without becoming score, rank, or control,
- how such a system coexists with existing legal, monetary, and political structures without replacing them.

This book does not describe a civilization. It describes a protocol—a pattern of operation that can be instantiated locally, partially, or temporarily, independent of global adoption.

2. What This System Is Not

The following exclusions are explicit.

This system is not:

- a replacement for nation-states,

- a substitute for legal systems,
- a proposal to abolish money,
- a moral scoring mechanism,
- a reputation or credit system,
- a social ranking structure,
- a governance model,
- a theory of justice.

No chapter should be read as implying otherwise. Where these structures exist, this system operates beside them.

3. Inherited Assumptions

This book does not restate its philosophical premises. They function as axiomatic constraints, not arguments.

The following are inherited from prior work and not re-defended:

Irreducibility of Persons Individuals are not interchangeable. Systems that treat them as such violate the precondition.

Non-Violation Force and coercion destroy the conditions of relationship. They cannot serve as system primitives.

Consent as Sole Basis of Connection Interaction must be voluntarily entered and voluntarily exited.

Promise as Fundamental Relational Form Contracts enforce behavior. Promises preserve relationship. This system operates on the latter.

Trust Is Not Reputation Trust emerges from lived interaction. It is not an assigned score.

Value Cannot Be Compressed Without Loss Single-scalar representations erase dimensions that matter.

These are not defended here. They are the boundaries within which this book operates.

4. Definition of Calculation

The term *calculation* is used narrowly.

It does not mean:

- maximizing a scalar objective,
- ranking persons or actions globally,
- producing a universal measure of worth.

It means only:

- determining whether declared conditions have been met,
- comparing outcomes within a shared context,
- supporting local, voluntary decision-making.

Accordingly:

- value representations are multi-dimensional,
- no operation produces a mandatory global ordering,
- aggregation is local, contextual, and optional,
- refusal is a first-class outcome, not a failure state.

Calculations that violate these constraints fall outside the protocol.

5. Non-Goals and Prohibitions

The following are explicitly forbidden, regardless of efficiency or convenience:

- global scores,
- universal rankings,
- fungible tokens derived from promise outcomes,
- automatic enforcement or punishment,
- irreversible accumulation of advantage,
- mandatory transparency of personal data,
- conversion of relational history into transferable capital.

Any implementation drifting toward these is no longer an implementation of this protocol.

6. Parallel Coexistence

This system assumes the continued existence of:

- money-based markets,
- legal enforcement mechanisms,
- institutional governance.

It does not compete with them.

Participation is opt-in. Non-participation incurs no penalty. Exit is always permitted.

The system is designed to fail gracefully—by becoming irrelevant, not coercive.

7. How to Read This Book

Each part answers one technical question:

- **Part I** — Why formalization is necessary when promises scale.
- **Part II** — How a promise becomes a formal object.
- **Part III** — How value is recorded without reduction.
- **Part IV** — How exchange occurs without coercion.
- **Part V** — How trust emerges without becoming score.
- **Part VI** — Where the system ends and what lies beyond.

No part justifies another philosophically. If the inherited assumptions are rejected, the system should be rejected entirely.

This book proceeds under one question only:

Given these constraints, can a system still operate?

What follows is the answer.

Technical Preface — End

Part I — From Recognition to Formalization

Why Human Promises Break at Scale

Chapter 1

The Limits of Human Memory and Attention

A promise between two people who know each other requires no external apparatus.

When A promises B to return a borrowed tool by Friday, both parties hold the commitment in memory. A knows what was promised. B knows what to expect. If A fails, B notices. If B forgets, A can remind. The promise persists because both participants maintain continuous awareness of it. This configuration has a ceiling.

Human working memory operates within narrow constraints. Cognitive research consistently identifies limits on the number of distinct commitments, relationships, and contextual details a person can actively track. These limits are not cultural. They are structural features of human cognition.

As the number of participants increases, the number of possible promise-relationships grows faster than any individual's capacity to monitor them. In a group of five, there are ten possible pairwise commitments. In a group of fifty, there are over a thousand. No participant can hold all of them in active awareness.

This is not a failure of intention. It is a constraint of architecture.

When promises exceed the threshold of direct cognitive access, several things occur:

- Participants begin to forget commitments they made.
- Participants begin to misremember what was promised.
- Participants lose track of whether a promise was fulfilled.
- Disputes arise not from dishonesty, but from divergent recollection.

None of these outcomes require bad faith. They emerge from the mismatch between the volume of commitments and the capacity to hold them.

Historically, this problem was addressed through externalization: written

records, witnesses, institutions, enforcement mechanisms. Book A has already examined what these solutions cost—the displacement of trust, the introduction of coercion, the conversion of relationship into procedure.

This chapter does not revisit that critique.

It observes only this: human recognition, unaided, does not scale. Beyond a certain density of commitment, promises begin to dissolve—not because people stop caring, but because the cognitive infrastructure required to preserve them does not exist.

The question that follows is not whether externalization is good or bad. The question is whether there exists any alternative to dissolution when memory fails.

That question remains open.

Chapter 2

Time, Drift, and the Loss of Shared Context

Even where memory persists, time introduces a second failure mode.

A promise is made at a particular moment, under particular conditions, with a particular understanding shared between parties. That shared understanding is not static. It changes as circumstances change, as language shifts, as the parties themselves change.

Consider: A promises B to "help when needed." At the moment of the promise, both A and B share an implicit model of what "help" means, what "needed" refers to, and under what conditions the promise applies. This shared model is not written. It exists as tacit alignment.

Six months later, that alignment may no longer hold.

B's situation has changed. What once counted as "needed" no longer applies. A's capacity has shifted. The meaning of "help" has drifted.

Neither party has violated the promise. But the promise no longer refers to the same commitment it once did.

This is contextual drift.

Drift does not require deception. It does not require negligence. It is an inevitable consequence of the fact that promises are made in time but

must be fulfilled across time. The context that gave a promise its meaning does not travel with the promise. It remains anchored to the moment of agreement.

When drift occurs:

- The promiser may believe the promise was fulfilled; the recipient may disagree.
- The recipient may believe the promise was violated; the promiser may be unaware.
- Both parties may hold the promise in good faith while referring to incompatible commitments.

In small-scale, high-contact relationships, drift can be corrected through ongoing negotiation. Parties revisit the promise, update their understanding, realign their expectations. This works when interaction is continuous and communication is low-cost.

At scale, continuous renegotiation becomes impractical. The cost of maintaining alignment across many promises, many parties, and long durations exceeds the capacity of informal coordination.

The result is not conflict, but incoherence. Promises persist in name but lose their referent. They become symbols without shared meaning.

This is not a problem that goodwill can solve. It is a structural feature of commitments that exist in time.

Chapter 3

When a Promise Becomes Unjudgeable

A promise that cannot be remembered has already failed. A promise whose meaning has drifted may fail without either party knowing.

There is a third failure mode, distinct from both: the promise that cannot be judged.

Judgment here means determination of outcome. Did the promise get fulfilled? Was it violated? Is it still pending? These are binary or ternary questions. They require answers.

In small-scale settings, judgment is informal. Both parties observe the outcome. If disagreement arises, they negotiate. The judgment emerges

from the relationship itself.

At scale, informal judgment breaks down.

When a promise involves parties who do not share continuous contact, when the fulfillment conditions are ambiguous, when the time horizon is long, the question "was this promise kept?" may have no accessible answer. Not because evidence is hidden, but because the structure of the promise does not support determination.

Consider the following conditions that make a promise judgeable:

1. The parties are identifiable.
2. The content is specific enough to admit verification.
3. The time window is bounded.
4. The completion condition is observable.

If any of these conditions is absent, judgment becomes indeterminate.

A promise to "support you in your career" lacks a completion condition. A

promise to "deliver the project" without a deadline lacks a time window.

A promise made to "the community" lacks identifiable counterparties. A

promise whose content was never made explicit lacks verifiable scope.

These are not bad promises in a moral sense. They may be sincere, meaningful, and deeply felt. But they are unjudgeable. No procedure, internal or external, can determine their outcome.

This observation is critical.

Book A has established that coercion cannot be used to enforce promises

without destroying the conditions that make promises meaningful. If

coercion is excluded, judgment must emerge from the structure of the

promise itself. But if the promise lacks the structural features required

for judgment, then judgment cannot emerge at all.

The result is a dilemma:

- Without external enforcement, promises must be self-judging.
- But many promises, as ordinarily made, lack the features required for self-judgment.
- Therefore, such promises become unjudgeable—neither kept nor broken, merely unresolved.

This is not a problem of ethics. It is a problem of structure.

Part I concludes here.

The three failure modes—memory limits, contextual drift, and structural unjudgeability—are not independent. They compound. A promise made at scale, across time, without explicit structure, faces all three simultaneously.

The observation is complete: human promises, as ordinarily made, do not survive the conditions of scale, time, and structural ambiguity.

What remains is a question of possibility.

If promises are to function beyond the limits identified here, something must change—not in human nature, not in moral commitment, but in the structure of the promise itself.

What that structure might be is not addressed in this Part.

Part I — End

Part II — Promise as an Atomic Unit

Defining the Smallest Formal Commitment

Chapter 4

Why Transactions Are the Wrong Primitive

Most systems that attempt to record human coordination use the transaction as their fundamental unit.

A transaction is an exchange: one party transfers something to another, and the other transfers something back. The transfer is instantaneous or near-instantaneous. Once complete, the relationship between the parties is settled. There is no residue, no ongoing commitment, no future obligation.

This structure has properties that make it attractive for system design:

- It is bounded in time.
- It is symmetric in form.
- It is immediately verifiable.
- It leaves no open state.

These properties also make it inadequate for the problems identified in Part I.

The failures described there—memory limits, contextual drift, and structural unjudgeability—do not arise primarily from instantaneous exchanges. They arise from commitments that extend across time, involve asymmetric obligations, and require future action.

A promise to deliver a service next month is not a transaction. It is an open commitment with a future fulfillment condition.

A promise to support someone in a time of need is not a transaction. It is an indefinite obligation triggered by circumstances that have not yet occurred.

A promise to collaborate on a shared project is not a transaction. It is an ongoing relationship with distributed, evolving responsibilities.

When such commitments are forced into transactional form, distortion occurs.

The commitment must be converted into an immediate exchange—often

of money or tokenized equivalents. The future dimension is collapsed into present settlement. The asymmetry of obligation is flattened into symmetric transfer. The relational continuity is severed.

This conversion does not preserve the commitment. It replaces it with something else.

The transaction-based model also inherits a structural assumption: that all value can be expressed in a single, fungible medium. If a commitment cannot be priced, it cannot be transacted. If it cannot be transacted, it does not exist within the system.

This assumption excludes precisely the commitments that Part I identified as failing at scale—those that are temporally extended, contextually embedded, and resistant to single-metric valuation.

The problem is not that transactions are useless. They function adequately for a narrow class of coordination problems. The problem is that they have been treated as the universal primitive, applied to domains where their structural assumptions do not hold.

Part I established that promises fail at scale due to memory, drift, and unjudgeability.

Transactions do not address these failures. They avoid them by restricting the domain to commitments that do not extend, do not drift, and do not require future judgment.

If the goal is to preserve promises—not replace them—a different primitive is required.

Chapter 5

The Promise Atom

A promise, as ordinarily made, is informal. It exists in the understanding between parties, shaped by context, tone, and shared history. This informality is not a flaw. It is what makes promises relational rather than mechanical.

But informality is also what makes promises vulnerable to the failures identified in Part I.

The question is not how to eliminate informality. The question is: what is

the minimum formal structure a promise must have in order to remain judgeable?

This chapter defines that structure.

A promise becomes a formal object—a *Promise Atom*—when it possesses four components:

1. Scope

Scope defines what the promise is about.

A promise without scope is a gesture. It may express goodwill, but it cannot be evaluated. "I promise to help you" is not judgeable. "I promise to help you move your belongings on Saturday" is.

Scope does not require exhaustive specification. It requires sufficient specification to distinguish fulfillment from non-fulfillment.

2. Counterparty

A promise is made to someone.

An anonymous commitment, or a commitment to an indefinite group, lacks a party capable of recognizing fulfillment. If no one can determine whether the promise was kept, the promise is not judgeable.

Counterparty identification does not require legal identity. It requires that the recipient of the promise is determinate—that there exists some party to whom fulfillment is owed.

3. Time Window

A promise operates within a temporal boundary.

Without a time window, a promise cannot transition from "pending" to "fulfilled" or "unfulfilled." It remains indefinitely open, subject to the contextual drift described in Part I.

The time window need not be precise. "By the end of the month" is sufficient. "Someday" is not.

4. Completion Condition

A promise specifies what counts as fulfillment.

If the completion condition is unstated or ambiguous, the promise cannot be judged even if scope, counterparty, and time window are defined. "I promise to help you move on Saturday" requires a condition: when is the

promise complete? When the truck is loaded? When everything is in the new location? When you say so?

The completion condition is the criterion by which the promise transitions from Declared to Fulfilled or Unfulfilled.

These four components are necessary conditions. A commitment lacking any one of them is not a Promise Atom.

- Without Scope, there is nothing to fulfill.
- Without Counterparty, there is no one to whom fulfillment is owed.
- Without Time Window, there is no point at which judgment becomes possible.
- Without Completion Condition, there is no criterion for judgment.

This is not a claim about what promises should be. It is a structural observation: these are the minimum features required for a promise to be judgeable within any system that does not rely on external enforcement.

Chapter 6

States of a Promise

A Promise Atom, once formed, exists in one of four states.

These states are not evaluations. They are positions in a state space. A promise does not become "good" or "bad" by entering a particular state. It simply occupies that state until conditions change.

Declared

A promise enters the Declared state when it is made.

At this point, Scope, Counterparty, Time Window, and Completion Condition are specified. The promise exists as a formal object. It has not yet been fulfilled. The time window has not yet closed.

All Promise Atoms begin in the Declared state.

Fulfilled

A promise transitions from Declared to Fulfilled when the Completion Condition is met within the Time Window.

This transition is binary. Either the condition is met, or it is not. There is no partial fulfillment at the atomic level. A promise that is "mostly kept"

remains Declared until the condition is fully met, at which point it becomes Fulfilled.

Unfulfilled

A promise transitions from Declared to Unfulfilled when the Time Window closes without the Completion Condition being met.

This is not a punishment. It is a state. The promise was not kept within its specified bounds. The Unfulfilled state records this fact.

Withdrawn

A promise transitions from Declared to Withdrawn when both parties agree to dissolve the commitment before the Time Window closes.

Withdrawal is not the same as failure. It is mutual release. Both parties consent to end the promise without fulfillment and without marking it as unfulfilled.

Withdrawal requires consent from both Counterparties. Unilateral withdrawal is not possible within this structure. A party who abandons a promise without consent does not withdraw it; the promise remains Declared until the Time Window closes, at which point it becomes Unfulfilled.

The four states form a closed set:

- Declared → Fulfilled (condition met)
- Declared → Unfulfilled (time window closed, condition not met)
- Declared → Withdrawn (mutual consent)

No other transitions are defined. A Fulfilled promise does not become Unfulfilled. An Unfulfilled promise does not become Fulfilled. A Withdrawn promise does not reactivate.

This state model is minimal. It captures only what is necessary for judgment: whether a promise was made, whether it was kept, and if not, why not.

It does not capture:

- The value of the promise.
- The effect of fulfillment or non-fulfillment.

- The relationship between this promise and others.
- The consequences for future interaction.

Those dimensions exist. They are addressed elsewhere.

Here, the only claim is structural: a Promise Atom can be in one of four states, and the transitions between them are determinate.

Part II — End

Part III — Valuation Without Reduction

Recording Change Without Compressing Value

Chapter 7

Why Scalar Value Collapses Meaning

A scalar is a single number.

When a system represents something with a scalar, it asserts that the thing can be fully captured by one dimension. Height is a scalar.

Temperature is a scalar. These work because height and temperature are, by definition, one-dimensional quantities.

Human commitments are not one-dimensional.

A promise kept may produce relief, or material improvement, or restored relationship, or expanded possibility, or some combination of these.

These outcomes are not commensurable. They do not share a common unit. There is no exchange rate between emotional restoration and material gain, between expanded freedom and reduced uncertainty.

When a system forces such outcomes into a single number, it does not measure them. It erases them.

This erasure is not incidental. It is structural.

A scalar representation requires a conversion function—a rule that transforms heterogeneous inputs into a single output. This function must make choices: which dimensions matter, how much each dimension contributes, what weights apply. These choices are not neutral. They encode a particular perspective on what counts.

Once encoded, the perspective becomes invisible. The number appears objective. The conversion function disappears behind the output. Users of the system see only the result, not the compression that produced it.

This is how money operates.

Money converts labor, creativity, risk, time, skill, and relationship into a single figure. The conversion is lossy. Dimensions that do not fit the function are discarded. What remains is comparable, but comparison comes at the cost of fidelity.

The same structure appears in:

- Credit scores, which compress financial behavior into a single index.
- Performance ratings, which collapse multidimensional contribution into a grade.
- Reputation systems, which reduce relational history to a number.
- Social rankings, which flatten complex standing into a position.

In each case, the scalar enables comparison. And in each case, comparison requires the elimination of dimensions that resist aggregation.

This is not a moral failure. It is a design constraint.

A system that uses scalar representation must accept that it cannot preserve what it measures. The act of measurement, in this form, is also an act of destruction.

The question is whether measurement requires this form.

If all calculation demands scalar reduction, then the losses are unavoidable. If calculation can proceed without reduction, then a different structure becomes possible.

This chapter does not answer that question. It establishes only the cost: scalar representation collapses meaning. Whatever is gained in comparability is lost in dimension.

Chapter 8

Δ State: Recording Change, Not Worth

The previous chapter identified the structural cost of scalar valuation.

This chapter introduces an alternative object of calculation.

The object is not value. It is change.

A Promise Atom, as defined in Part II, transitions between states: Declared, Fulfilled, Unfulfilled, Withdrawn. Each transition marks a change. Something that was pending is now complete. Something that was open is now closed.

This change is observable. It does not require interpretation. The promise was in one state; now it is in another.

What the change *means*—what it is *worth*—is a different question. That question invokes comparison, preference, judgment. It requires a framework external to the promise itself.

But the change itself requires no such framework. It is a fact about the promise's trajectory.

This is Δ State.

Δ State is defined as follows:

The recorded difference in condition that results from the transition of a Promise Atom from one state to another.

Δ State is:

Relative. It describes change, not position. A Δ State of "fulfilled" does not indicate how much was gained. It indicates that what was Declared is now Fulfilled.

Local. It applies to the parties involved in the promise. It does not generalize. A's Δ State from a promise with B says nothing about A's Δ State with C.

Contextual. Its meaning depends on the scope of the promise. A fulfilled promise to deliver a report and a fulfilled promise to provide emergency assistance both produce Δ State = Fulfilled. The contexts are incommensurable. The state transition is the same.

Δ State is not a score.

A score implies ranking. Δ State does not rank. Two fulfilled promises both register Δ State = Fulfilled. There is no mechanism within Δ State to determine which fulfillment was "better."

Δ State is not a measure of worth.

Worth implies a scale. Δ State provides no scale. It records that a transition occurred, not how valuable that transition was.

Δ State is not fungible.

Fungibility implies substitutability. One unit can replace another. Δ State does not admit substitution. The Δ State from Promise X cannot be exchanged for the Δ State from Promise Y. They are distinct records of

distinct transitions.

What Δ State does permit:

Recording. A system can store that a transition occurred, when it occurred, and between which parties.

Accumulation. Over time, a party may have multiple Δ State entries. These entries can be collected. They form a history.

Local comparison. Within a shared context—between the same parties, or within the same domain— Δ State entries can be compared. More fulfilled promises, fewer unfulfilled promises. This comparison is bounded. It does not extend beyond the context.

Δ State answers a narrow question: what changed?

It does not answer: how much is it worth? It does not answer: how does this compare globally? It does not answer: what should happen next?

Those questions remain open.

Chapter 9

Multi-Dimensional Accumulation

Δ State, as defined, is local and contextual. It does not aggregate into a global measure. This is not a limitation to be overcome. It is a design constraint that preserves the structure established in earlier chapters. This chapter addresses the question: if global aggregation is excluded, what kind of accumulation remains possible?

Consider two parties, A and B, who have made multiple promises to each other over time. Each promise has transitioned through states. Some were fulfilled. Some were not. Some were withdrawn.

This history can be represented as a collection of Δ State entries:

- Promise 1: Fulfilled
- Promise 2: Fulfilled
- Promise 3: Unfulfilled
- Promise 4: Withdrawn

- Promise 5: Fulfilled

This collection is not a number. It is a record. It has structure, but the structure is not scalar.

The collection can be queried:

- How many promises were fulfilled? (3)
- How many were unfulfilled? (1)
- What is the ratio? (3:1)

These queries produce numbers, but the numbers do not reduce the collection. They describe aspects of it. The collection itself remains intact, with its individual entries preserved.

Now consider that promises vary along multiple dimensions:

- Domain (material, informational, relational, temporal)
- Scope (narrow, broad)
- Duration (short-term, long-term)
- Complexity (simple, compound)

Each dimension is independent. A promise may be narrow in scope but long in duration. Another may be broad in scope but short in duration. These promises are not comparable along a single axis.

When Δ State entries are accumulated, they can be organized by dimension. The result is not a single aggregate, but a structure:

- Material promises: 2 fulfilled, 0 unfulfilled
- Informational promises: 1 fulfilled, 1 unfulfilled
- Relational promises: 0 fulfilled, 0 unfulfilled

This is a vector, not a scalar. It has components. The components do not collapse into a sum.

A vector structure permits partial ordering.

In a partial order, some elements are comparable and some are not.

If A has (3 fulfilled, 1 unfulfilled) and B has (2 fulfilled, 2 unfulfilled), a limited comparison is possible: A has more fulfillments and fewer non-fulfillments. A is "greater" in this partial sense.

But if A has (3 fulfilled, 1 unfulfilled) and C has (2 fulfilled, 0 unfulfilled),

no total ordering exists. A has more fulfillments, but C has fewer non-fulfillments. Neither dominates the other.

Partial ordering does not produce a ranking. It produces a structure of comparabilities and incomparabilities. Some pairs can be compared. Others cannot.

This structure has consequences:

No global leaderboard. Without total ordering, there is no way to rank all parties from best to worst. The question "who is highest?" has no answer.

No fungibility. Because dimensions are preserved, Δ State entries cannot be traded across dimensions. Three material fulfillments do not convert into one relational fulfillment.

No equivalence. Two parties with identical scalar sums may have entirely different vector profiles. The sum conceals; the vector reveals.

What remains possible:

Contextual queries. Within a specific dimension or relationship, questions can be answered. "How has A performed in material promises with B?" is answerable. "How has A performed overall?" is not—because "overall" implies a reduction that the structure forbids.

Pattern recognition. Over time, profiles develop. A party with consistent fulfillment in one domain and consistent non-fulfillment in another exhibits a pattern. The pattern is observable without being scored.

Local decision support. When a party considers entering a new promise with another, the accumulated Δ State record provides information. Not a recommendation. Not a score. Information, structured by dimension and context.

This information is input for human judgment. The system does not act on it. No threshold triggers automatic consequence. No pattern produces mandatory response. The comparison exists to inform, not to compel.

What a party does with this information is outside the system's scope.

This chapter does not claim that multi-dimensional accumulation is superior, fairer, or more accurate than scalar aggregation. Such claims would require a framework of evaluation that this chapter does not provide.

The claim is narrower: accumulation without reduction is structurally possible. Δ State can be recorded, collected, and compared without collapsing into a single number.

Whether this structure is useful, and for what, is not addressed here.

Part III — End

Part IV — Exchange Without Coercion

How Interaction Occurs Without Control

Chapter 10

Why Enforcement Is a Systemic Shortcut

Enforcement is a mechanism that guarantees outcome.

When a system includes enforcement, it ensures that commitments produce their specified results regardless of whether the parties continue to will them. A contract backed by legal remedy will be fulfilled—or compensated—whether or not the obligor cooperates. The enforcement layer removes the uncertainty.

This removal has structural consequences.

Enforcement operates by substituting external force for internal continuation. When A promises B and later fails to act, enforcement provides B with recourse that does not depend on A's cooperation. The court, the collateral, the penalty clause—these mechanisms bypass A's will and deliver an outcome.

From a system design perspective, this is efficient. Uncertainty is expensive. Verification is costly. Enforcement compresses the space of possible outcomes into a narrower, more predictable range.

This is what enforcement solves: it reduces variance.

What enforcement destroys is less visible but equally structural.

Voluntariness. Once enforcement exists, the decision to fulfill a promise is no longer fully voluntary. The party acts under constraint. The fulfillment may occur, but its character changes. It is no longer an expression of continued commitment. It is compliance under pressure.

Relational residue. A promise kept under enforcement leaves no relational trace beyond the transaction itself. The interaction is closed. Nothing carries forward. The parties exit as they entered—connected by obligation, not by choice.

Meaning of refusal. In an enforced system, refusal is not a state. It is a

violation. The party who does not perform is not registering a choice; they are failing a requirement. The system does not recognize "no" as a valid output.

Enforcement is not irrational. It exists because it works. It enables coordination at scale among parties who do not know or trust each other. It is the mechanism that permits strangers to transact.

But enforcement is also a shortcut. It achieves coordination by removing the conditions under which coordination would need to be chosen. It solves the problem of uncertain commitment by eliminating the space in which commitment is uncertain.

This chapter does not argue that enforcement is wrong. It observes that enforcement is a specific structural choice with specific structural costs. Systems that include enforcement cannot simultaneously preserve voluntariness, relational continuity, and the meaningfulness of refusal. Whether those costs are acceptable depends on what the system is for.

Chapter 11

Voluntary Matching

If enforcement is excluded, what remains?

The answer is matching—but matching of a specific kind.

A match occurs when two parties each opt in to the same interaction. This is not a contract. A contract specifies obligations and binds parties to performance. A match specifies only that both parties have expressed willingness to proceed. What happens next depends on continued willingness, not on prior commitment.

Matching has the following structural features:

Explicit opt-in. A match does not occur by default. Neither party is included unless they affirmatively signal participation. Silence is not consent. Inaction is not agreement.

Symmetry. Both parties must opt in. A unilateral declaration does not create a match. If A opts in and B does not, no interaction occurs. The

system does not override B's non-participation.

Revocability. Opt-in can be withdrawn before the interaction completes. Either party may exit. The system does not penalize withdrawal. It records it.

No future binding. A match does not create obligation beyond itself. Participation in one interaction does not commit a party to future interactions. Each match is independent.

Under these conditions, exchange occurs only when both parties choose it, and only for as long as both parties continue to choose it.

The system permits matching. It does not guarantee it.

If A wants to interact with B and B does not opt in, the interaction does not occur. The system does not mediate, persuade, or incentivize. It records A's expressed interest and B's absence of response. The state is: no match.

If both parties opt in and later one withdraws, the interaction terminates. The system does not complete the interaction on behalf of the withdrawing party. It records the withdrawal. The state is: match terminated.

This structure has a consequence: not all desired interactions will occur. A party seeking a match may not find one. A party offering something may receive no uptake. The system does not correct for this. It does not optimize matching rates. It does not suggest alternatives.

What the system does is narrower: it records expressed willingness, detects mutual overlap, and marks the resulting state. Match or no match. Ongoing or terminated.

Exchange, in this framing, is not a transaction completed. It is a state entered. The state persists as long as both parties remain opted in. When either exits, the state ends.

Chapter 12

Refusal, Exit, and Non-Participation

A system without enforcement must define what happens when parties do not participate.

In enforced systems, non-participation is anomalous. It triggers remedy, penalty, or escalation. The system is designed to produce participation; non-participation is a failure state.

In the structure described here, non-participation is not a failure. It is a first-class outcome.

Refusal.

Refusal is the state in which a party declines to opt in.

This is not an error. It is not recorded as a deficiency. The system does not distinguish between "refused" and "not interested" and "not available." It records only: this party did not opt in.

Refusal carries no consequence within the system. A party who refuses one interaction is not marked, downgraded, or flagged. Their future options are not constrained by past refusals.

This is structural, not aspirational. If refusal carried consequence, refusal would not be free. And if refusal is not free, opt-in is not voluntary.

Exit.

Exit is the state in which a party withdraws from an interaction already in progress.

A match, once formed, may be exited by either party at any time before completion. The system records the exit. It does not interpret the exit. It does not assign cause or blame.

Exit does not trigger penalty. A party who exits is not charged, fined, or restricted. Their record reflects that they exited. Nothing more.

If exit carried penalty, the structure would reintroduce enforcement through the back door. The threat of penalty would constrain the decision to exit. Exit would no longer be free. And if exit is not free, participation is not voluntary.

Non-participation.

Non-participation is the state in which a party does not engage with the system at all.

This is also a valid state. The system does not require participation. It does not assume that all eligible parties will or should participate. It does not measure participation rates or treat low participation as a problem to be solved.

A party outside the system is not deficient. They are simply outside.

These three states—refusal, exit, non-participation—are necessary conditions for a non-coercive structure.

If refusal is penalized, the system coerces participation. If exit is penalized, the system coerces continuation. If non-participation is stigmatized, the system coerces entry.

A system that excludes enforcement must therefore treat all three as neutral outcomes. Not preferred. Not discouraged. Neutral.

This has a structural implication: the system cannot guarantee activity.

If all parties refuse, nothing happens. If all parties exit, interactions collapse. If no one participates, the system is empty.

The system does not prevent these outcomes. It permits them.

This is not a flaw. It is the cost of non-coercion. A system that guarantees activity must compel it. A system that refuses to compel must accept that activity may not occur.

Part IV — End

Part V — Trust as an Emergent Property

How Trust Affects the System Without Becoming a Score

Chapter 13

Why Trust Cannot Be Recorded Directly

Trust is not data.

Data can be stored, retrieved, and compared. It persists independently of the parties who generated it. It can be transmitted without loss. Trust has none of these properties.

When A trusts B, the trust exists in A's orientation toward B. It is not an object that can be extracted from A, stored in a system, and later queried. It has no location outside the relationship in which it occurs.

Trust is not a state variable.

A state variable is a system-readable property that changes according to defined rules. Part II introduced Promise states: Declared, Fulfilled, Unfulfilled, Withdrawn. These are state variables. They can be read by the system. They transition according to specified conditions.

Trust does not behave this way. There is no operation that sets trust to a value. There is no condition that transitions trust from one state to another. Trust is not readable by the system because it does not exist as a system object.

Trust is not an accumulable asset.

Assets can be counted, summed, and transferred. If A has three units of something and gains two more, A has five. Trust does not admit this arithmetic. There is no unit of trust. There is no sum. There is no transfer.

A cannot give their trust in B to C. A cannot spend trust. A cannot convert trust into another form.

Three distinctions clarify what trust is not:

Trust is not reputation. Reputation is an external assignment. It is

what others say about a party. It can be aggregated, published, and compared. Trust is internal to the one who trusts. It cannot be aggregated across observers without becoming something else.

Trust is not compliance. Compliance is adherence to rules. A party may comply perfectly and still not be trusted. A party may be trusted despite imperfect compliance. The two vary independently.

Trust is not reliability. Reliability is a pattern of behavior. It can be observed and recorded. A party who fulfills promises consistently exhibits reliability. But reliability does not produce trust automatically. A party may observe another's reliability and still withhold trust. The inference is not guaranteed.

Any attempt to record trust directly transforms it.

If a system assigns a value to trust, that value becomes the object of optimization. Parties begin to act in ways that increase the recorded value rather than in ways that would generate trust. The record replaces the phenomenon.

If a system publishes trust, trust becomes reputation. The internal orientation is externalized, aggregated, and made comparable. It is no longer trust.

If a system enforces consequences based on recorded trust, trust becomes compliance. The party acts to avoid penalty or gain advantage, not because they are oriented toward the other.

This is not a recommendation against recording trust. It is an observation: trust, as a phenomenon, does not survive the operation of direct recording. What gets recorded is always something else.

The question that follows is whether a system can be affected by trust without recording it.

Chapter 14

τ as Potential, Not Measure

If trust cannot be recorded directly, how can a system reflect its presence

or absence?

The answer requires a different kind of variable.

τ (tau) is introduced here not as a measure of trust, but as a structural property that constrains what interactions remain available.

τ is not a score. There is no scale from low to high. There is no comparison between one party's τ and another's.

τ is not a ranking. Parties are not ordered by τ . There is no leaderboard, no percentile, no relative position.

τ is not an evaluation. The system does not judge parties based on τ . No inference is drawn about character, worth, or deservingness.

τ is not a quantity of trust. It does not represent "how much" trust exists. Trust is not quantifiable; τ does not quantify it.

What τ does:

τ adjusts the range of interactions that remain structurally available to a party.

This adjustment is negative, not positive. τ does not grant access. τ does not unlock capability. τ does not open doors.

τ removes obstacles. When τ is in a certain condition, certain friction is absent. When τ is in another condition, certain friction is present.

The metaphor is not a key that opens locks. The metaphor is a wall that may be lower or higher. The party still chooses whether to approach. The wall does not move them.

τ is not directly modified.

There is no operation that increases τ . There is no operation that decreases τ . The system does not act on τ .

τ is shaped indirectly by the pattern of Δ State entries over time.

When a party's interactions consistently result in Fulfilled states, the pattern is visible. When a party's interactions frequently result in

Unfulfilled states, that pattern is also visible. These patterns do not "add up" to a τ value. They constitute the conditions under which τ takes its

current form.

τ does not create thresholds.

There is no point at which τ crosses a line and triggers a consequence.

There is no automatic gate. There is no "minimum τ required."

τ affects the structural conditions of interaction, but never determines outcome. A party with any τ condition may still attempt any interaction. The interaction may face more friction or less. But the system does not prohibit.

τ has three structural features:

Contextual visibility. τ is not global. It exists relative to specific domains, relationships, or interaction types. A party's τ in one context says nothing about their τ in another. There is no aggregate τ .

Decay. τ conditions are not permanent. Over time, without reinforcing patterns, the structural effects of τ diminish. This is not punishment. It is the natural dissipation of pattern-based influence. Past interactions become less structurally relevant as they recede.

Recovery. τ conditions that impose friction are not permanent either. When patterns change, the conditions change. This cannot be forced. A party cannot perform an action to "restore" τ . Recovery occurs through the gradual establishment of different patterns over time.

τ does not reward. τ does not punish. τ does not recommend.

τ is the structural residue of interaction patterns, visible only in the friction or ease that future interactions encounter.

Chapter 15

System Effects of Accumulated Trust

If τ affects interaction conditions, what specifically changes?

This chapter describes the structural effects—not as rewards or penalties, but as shifts in what the system requires or permits.

The question is not: what does a party gain when trust accumulates?
The question is: what does the system stop doing?

Reduced verification requirements.

In the absence of prior interaction, the system may require explicit confirmation at multiple stages. Both parties must affirm, re-affirm, and confirm completion.

When Δ State patterns between two parties reflect consistent fulfillment, these verification steps may become unnecessary. Not because the parties have "earned" reduced scrutiny, but because the structural conditions that required scrutiny are no longer present.

The system does not decide to trust. It simply has fewer reasons to interrupt.

Expanded interaction range.

Certain interactions may be structurally unavailable when parties have no shared history. Not prohibited—but practically inaccessible due to the friction involved.

When patterns develop, this friction may reduce. Interactions that were structurally cumbersome become structurally simpler. Not because permission is granted, but because obstacles are absent.

The system does not approve. It simply does not obstruct.

Reduced cost of matching failure.

When a match fails—when a party exits, or a promise goes unfulfilled—there is a structural residue. Future interactions may carry more friction. When prior patterns reflect consistent fulfillment, a single failure produces less structural effect. The residue is smaller. Not because the failure is excused, but because it occurs against a background that absorbs it.

The system does not forgive. It simply does not amplify.

What the system does not do:

The system does not grant rewards. There is no benefit conferred for accumulated trust.

The system does not assign priority. No party receives preferential access.

The system does not automate approval. No interaction is pre-authorized based on τ .

The system does not create privilege. No party gains structural advantage over others.

The system does not differentiate treatment based on τ as a category. There are no "trusted" and "untrusted" classes.

The asymmetry is deliberate.

Trust does not open doors. Trust lowers walls.

A party whose patterns reflect consistent fulfillment does not gain the ability to do things others cannot. They encounter less friction in doing what anyone can attempt.

The positive space—what becomes possible—is unchanged. The negative space—what obstacles exist—is reduced.

This is the structural limit of what the system can reflect.

The system does not generate trust. It cannot. Trust emerges from lived interaction, from choices made over time, from the accumulation of experience that no system can substitute.

What the system can do is survive in conditions where trust exists.

Where trust does not emerge, friction accumulates. Interactions become cumbersome. Patterns of non-fulfillment generate structural drag. The system does not collapse—but it also does not thrive.

Where trust does emerge, friction dissipates. Interactions flow. The system becomes lighter, not because it rewards, but because it does not need to obstruct.

The system does not produce trust. It persists in environments where trust has emerged. If trust does not emerge, the system remains possible—but structurally burdened.

Part V — End

Part VI — System Boundaries & Parallel Coexistence

Where the Engine Stops

Chapter 16

Opt-In by Design

Participation in this system is not required.

This is not a feature. It is a structural premise. The system does not function unless participation is voluntary, and it does not attempt to make participation attractive, convenient, or advantageous.

Opt-in as precondition.

Every interaction described in Parts II through V assumes that parties have chosen to engage. The Promise Atom requires explicit declaration. Matching requires mutual opt-in. Exit requires no justification.

These are not permissions granted by the system. They are the conditions under which the system operates at all. If participation were compelled, the structures described earlier would not hold. Voluntary matching becomes mandatory assignment. Exit without penalty becomes exit with consequence. The system would no longer be the system described. Opt-in is not added to the design. The design presupposes it.

Non-participation as normal state.

A party who does not participate is not in a failed state. They are in no state with respect to the system. The system does not register them, track them, or record their absence.

Non-participation is not:

- A deficit to be corrected.
- A status to be changed.
- A condition the system monitors.

The system has no model of "potential participants." It recognizes only those who have opted in. Everyone else is outside its scope.

Exit as equivalent to non-participation.

A party who exits returns to non-participation. The transition carries no residue. There is no record of "former participant." There is no category of "departed."

Exit and non-participation are structurally identical from the system's perspective. Both result in the same state: absence.

What the system does not track.

The system does not measure:

- Participation rate.
- Activity level.
- Retention.
- Growth.

These metrics assume that participation is a goal. The system has no such goal. It does not prefer more participants to fewer. It does not prefer active participants to inactive ones.

The system operates if parties opt in. It does not operate if they do not. Both outcomes are structurally equivalent from the system's perspective.

Three structural clarifications:

Non-participation $\neq$ failure. The system does not fail when parties do not participate. It simply does not operate for those parties.

Exit $\neq$ violation. Leaving the system is not a breach. It is a state transition to non-participation. No record of wrongdoing is created.

Silence $\neq$ consent. The absence of refusal is not opt-in. Participation requires affirmative action. Inaction produces non-participation, not participation.

Chapter 17

No Override, No Replacement

This system operates alongside existing structures. It does not compete with them, supersede them, or claim authority over them.

What the system does not replace.

The system does not replace legal enforcement. Courts, contracts, and statutory obligations continue to operate. The system has no mechanism to invalidate a legal judgment, void a contract, or override a regulatory requirement.

The system does not replace monetary exchange. Currency-based markets continue to function. The system does not issue currency, denominate value in monetary terms, or interact with financial infrastructure.

The system does not replace state authority. Governments retain their jurisdictional powers. The system does not claim sovereignty, recognize sovereignty, or adjudicate sovereignty.

The system does not replace contract-based coordination. Parties may continue to use contracts for any purpose. The system does not declare contracts inferior, obsolete, or unnecessary.

What the system cannot do.

The system cannot override external decisions. If a court orders enforcement, the system has no counter-mechanism. If a contract specifies a penalty, the system does not nullify it. External structures operate according to their own logic. The system does not intervene. The system cannot produce authoritative outcomes. Nothing generated by the system—no Δ State record, no τ condition, no match status—has legal standing, financial value, or political weight unless external structures independently choose to recognize it. The system does not confer authority.

The system cannot invalidate external records. If an external database records a transaction, the system does not contradict it. If a government registry lists a status, the system does not dispute it. The system's records exist in parallel. They do not intersect with external records unless parties and external structures both elect intersection.

Structural separation.

The system and existing structures occupy different operational layers.

Existing structures operate through enforcement, legal recognition, and institutional authority. The system operates through voluntary participation, pattern-based τ conditions, and structural friction. These layers do not conflict because they do not interact unless parties bring them into contact. A party may participate in both. A party may participate in one and not the other. The system does not require coherence between layers.

No claim of superiority.

The system does not claim to be better than existing structures. It does not claim to solve problems that existing structures fail to solve. It does not position itself as an alternative that parties should prefer. The system exists. Existing structures exist. Both continue.

Chapter 18

Graceful Failure

Systems fail. This system is designed to fail in a specific way.

Failure mode: irrelevance.

When activity decreases—when fewer parties opt in, when fewer promises are declared, when fewer matches form—the system does not respond.

It does not:

- Intensify outreach.
- Modify rules to encourage participation.
- Lower barriers to entry.
- Create incentives.
- Issue warnings.

The system does nothing. Activity decreases, and the system becomes less relevant. If activity ceases entirely, the system becomes irrelevant entirely.

This is the intended failure mode. The system fails by becoming unused, not by collapsing, fragmenting, or causing harm.

No self-preservation logic.

The system does not attempt to sustain itself.

There is no minimum activity threshold below which emergency measures activate. There is no resource pool dedicated to system continuation. There is no governance mechanism empowered to save the system.

If the conditions for operation cease to exist, operation ceases. The system does not struggle against this outcome.

No escalation pathway.

When interactions fail—when promises go unfulfilled, when matches dissolve, when τ conditions accumulate friction—the system does not escalate.

It does not:

- Impose additional requirements.
- Restrict future participation.
- Invoke external authority.
- Punish parties.

The system records what occurred. The structural effects described in Part V may apply. But there is no escalation. A failed interaction is a failed interaction. It does not trigger a sequence of increasingly severe responses.

No minimum requirements.

There is no minimum number of participants required for the system to operate. There is no minimum number of promises required. There is no minimum activity level.

The system operates at whatever scale parties choose. If that scale is zero, the system is dormant. If that scale is one interaction, the system operates for that interaction.

The system does not define success in terms of scale. It does not define failure in terms of inactivity.

What remains after failure.

If the system becomes entirely inactive, what remains is the specification itself—the structures described in Parts I through V.

These structures do not disappear because no one uses them. They remain as a description of how interaction could occur under the constraints identified.

Whether anyone chooses to interact under these constraints is not the system's concern. The system does not need to be used. It needs only to be coherent.

Terminal condition.

This chapter concludes Book B.

The system described in this book has defined:

- What it records (Part II).
- How it records without reduction (Part III).
- How interaction occurs without coercion (Part IV).
- How trust affects structure without becoming score (Part V).
- Where it stops (Part VI).

The system does not extend beyond these boundaries. It does not imply further development. It does not anticipate implementation. It does not promise results.

The specification is complete. The system is bounded.

What lies beyond the boundary is outside the scope of this document.

Part VI — End

Book B — End